



Práctica 4

Procesador: arquitectura, camino de datos y control

.....

| | |
|--------------------------------|--|
| Nombre y Apellidos | |
| Nombre y Apellidos | |
| Número de grupo de laboratorio | |

Preguntas 1 Traduzca las siguientes instrucciones RISC-V a lenguaje ensamblador: Un campo literal debe expresarse como entero en decimal, excepto en la instrucción “auipc” que se expresa como entero en hexadecimal.

| lenguaje máquina | 3322222222211111119876543210 | lenguaje ensamblador |
|--------------------------|------------------------------|----------------------|
| FF85AF03 | | |
| identificación de campos | | |
| campo literal: bits | | |
| entero | | |
| 00C735B3 | | |
| identificación de campos | | |
| 10000297 | | |
| identificación de campos | | |
| campo literal: bits | | |
| entero | | |

| | | |
|--------------------------|--|--|
| 00E780A3 | | |
| identificación de campos | | |
| campo literal: bits | | |
| entero | | |
| FEC79AE3 | | |
| identificación de campos | | |
| campo literal: bits | | |
| entero | | |
| FD1FF06F | | |
| identificación de campos | | |
| campo literal: bits | | |
| entero | | |

2 Un tipo numérico de datos en un lenguaje de alto nivel tiene un rango de valores limitado. En consecuencia, al efectuar operaciones algebraicas es posible que el resultado no se pueda representar (desbordamiento).

Suponga la operación de suma de los números naturales x e y , con un rango de valores $0 \leq x, y < M$, donde $M-1$ es el máximo valor que se puede representar. Por tanto, la operación $x + y$ produce desbordamiento cuando $x > (M-1) - y$.

Escriba una secuencia de instrucciones en lenguaje C que detecte desbordamiento sin efectuar la suma. Suponga que el valor máximo de un número natural está especificado por la constante MAX. En las operaciones que se especifiquen no debe producirse desbordamiento en ningún caso.

| secuencia de instrucciones C |
|---|
| int x, y; int desb; // 1 indica desbordamiento |
| |

Suponga que los operandos se representan con vectores de 32 bits ($M = \text{MAX} + 1 = 2^{32}$). Escriba una secuencia de instrucciones RISC-V en lenguaje ensamblador que detecte desbordamiento sin efectuar la suma. Tenga en cuenta que las operaciones de la secuencia no deben producir desbordamiento en ningún caso. El número de instrucciones debe ser el mínimo. Suponga que los operandos x e y están almacenados en los registros $x10$ y $x11$ respectivamente. El resultado se almacena en el registro $x9$ (el valor 1 indica desbordamiento).

| secuencia de instrucciones ensamblador | Justificación |
|--|---------------|
| | |

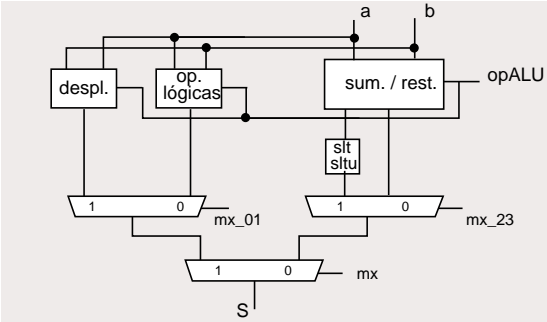
3 Suponga que el procesador interpreta una instrucción de secuenciamiento condicional que cumple la condición. Indique las instrucciones de secuenciamiento que pueden generar los valores de salida del módulo EVAL (“Secuenciamiento condicional relativo” en la página 246). Marque con X cualquier combinación que no se pueda producir.

| ig | me | meu | instruccion secuenciamiento condicional |
|----|----|-----|---|
| 0 | 0 | 0 | |
| 0 | 0 | 1 | |
| 0 | 1 | 0 | |
| 0 | 1 | 1 | |

| ig | me | meu | instruccion secuenciamiento condicional |
|----|----|-----|---|
| 1 | 0 | 0 | |
| 1 | 0 | 1 | |
| 1 | 1 | 0 | |
| 1 | 1 | 1 | |

En la siguiente pregunta se utiliza el diseño ubicado en el directorio PROC_SERIE_alu.

4 En la figura se muestran las 3 unidades funcionales de la ALU (página 250) y el árbol de multiplexores de selección: a) desplazamiento lógico o aritmético (a la derecha o a la izquierda), b) operación lógica, y c) sumador algebraico y comparador de menor (enteros o naturales). El módulo slt/sltu formatea (añade ceros a la izquierda) la salida de condición del sumador.

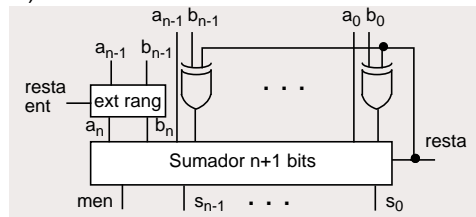


La ubicación de los ficheros relacionados con este proyecto (ALU.qpf) se indica en la página 253 (Trabajo, ubicación del directorio y subdirectorios). Las dos primeras unidades funcionales y el formateador ya están diseñados (ficheros despla.vhd, logica.vhd y slt.vhd respectivamente). El fichero ALU.vhd contiene la descripción estructural de la ALU, excepto la selección de las salidas de las 3 unidades y el formateador. Esta selección se efectúa mediante 2 niveles de multiplexores.

Deduzca las expresiones lógicas de las 3 señales de selección de los multiplexores en función de la señal de control opALU (“Señales de control de la ALU” en la página 289). Inclúyalas en cuerpo de la arquitectura.

| | | |
|---------|-------|--|
| nivel 1 | mx_01 | |
| | mx_23 | |
| nivel 2 | mx | |

El fichero sumalg.vhd contiene la interface del sumador algebraico y comparación de menor. Se utiliza un sumador de vectores de $n+1$ bits, puertas xor y lógica para extender el rango de los vectores a sumar. La salida men se activa cuando se cumple la condición $a < b$. Recuerde que los operandos se pueden interpretar como enteros o naturales. Esta salida se formatea (añadiendo ceros a la izquierda) en la unidad slt/sltu, que también está diseñada (fichero slt.vhd).



Analice el fichero sumalg.vhd. Deduzca las sentencias de asignación concurrente de las señales resta y ent (en función de opALU) y del bit más significativo de los vectores de entrada del sumador de $n+1$ bits (a_n , b_n). Escriba las sentencias de asignación de las salidas de la unidad (en función del vector resultado de la suma).

| | | | | |
|---------|-------|--|----|--|
| salidas | resta | | an | |
| | ent | | bn | |
| | men | | | |
| | s | | | |

Compruebe el diseño efectuado. El programa de prueba suministrado (prueba_ALU.vhd) compara, para un conjunto reducido de vectores, las salidas de la alu original y la diseñada¹.

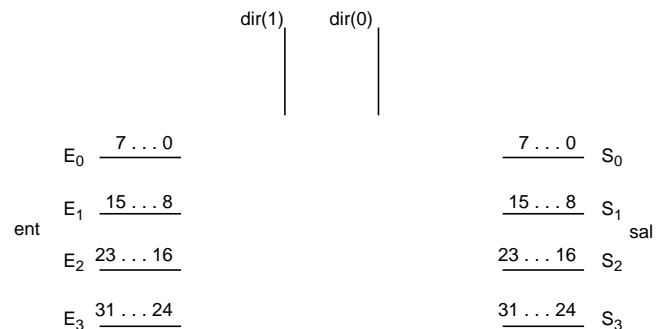
En las siguientes dos preguntas se utiliza el diseño ubicado en el directorio PROC_SERIE. El Apéndice 4.9 muestra la organización del proyecto en directorios.

1. Utilizando los directorios ubicados en el directorio PROC_SERIE_alu/ENSAMBLADO puede verificarse el funcionamiento del procesador cuando se incluye la ALU diseñada. Notemos que un programa usualmente sólo comprueba algunas operaciones. Efectúe esta comprobación después de responder a la última pregunta, una vez haya leído el mecanismo para efectuar una simulación ejecutando un programa..

- 5** El circuito FMTL (fichero FMTL.vhd) formatea el dato leído de la memoria que se utiliza para actualizar el banco de registros (“Segundo nivel: organización de la memoria de datos” en la página 254).

Analice el componente “alinear” (FMTL/ALINEAR/CODIGO) y muestre un esquema del circuito. Como ayuda, utilice el visor RTL de quartus (elabore el proyecto entero, ENSAMBLADO). Rellene la tabla que relaciona los datos de entrada y salida del módulo en función de los dos bits menos significativos de la dirección. Utilice la nomenclatura E_i y S_i para indicar el byte i del dato de entrada y de salida.

| dir 1..0 | sal | | | |
|-------------|-------|-------|-------|-------|
| | S_3 | S_2 | S_1 | S_0 |
| 0 0 | | | | E_0 |
| 0 1 | | | | E_1 |
| 1 0 | | | | E_2 |
| 1 1 | | | | E_3 |

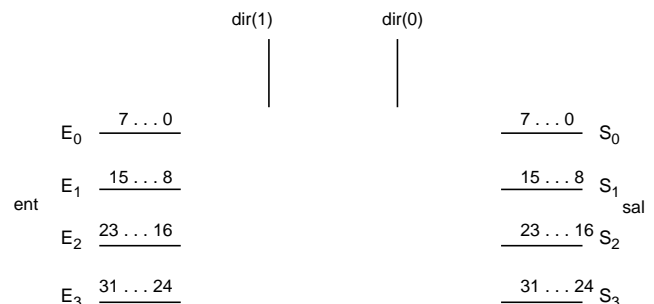


- 6** El circuito FMTE formatea el dato leído del banco de registros para actualizar los bancos de la memoria de datos (“Segundo nivel: organización de la memoria de datos” en la página 254).

En el diseño base (fichero FMTE.vhd), el circuito consta de 2 módulos: a) alineamiento de datos (fichero alinearE.vhd), y b) selección de los bancos a actualizar (fichero sel_byte.vhd). El módulo alinearE utiliza los 2 bits menos significativos de la dirección.

Analice la descripción de la arquitectura del módulo alinearE (FMTE/ALINEARE/CODIGO) y dibuje un esquema del circuito. Utilice el visor RTL de quartus (elabore el proyecto entero, ENSAMBLADO). Rellene la tabla que relaciona los datos de entrada y salida del formateador en función de los dos bits menos significativos de la dirección.

| dir 1..0 | sal | | | |
|-------------|-------|-------|-------|-------|
| | S_3 | S_2 | S_1 | S_0 |
| 0 0 | | | | E_0 |
| 0 1 | | | | E_0 |
| 1 0 | | | | E_0 |
| 1 1 | | | | E_0 |



7 Proponga un diseño alternativo del módulo alinearE que utilice únicamente los 2 bits menos significativos de la señal de control opMD para formatear el dato de escritura a memoria (“Figura 4.107” en la página 292). La ubicación de los ficheros relacionados con este proyecto (ALINEAE.qpf) se indica en la página 256. En primer lugar rellene la tabla que relaciona los datos de entrada y salida del formateador en función de los dos bits menos significativos de la dirección. Minimice el número de niveles de selección y el número de multiplexores de 2 entradas.

| | | | | |
|-----|---------------------------|--|---------------------------|-----|
| | opMD(1) | | opMD(0) | |
| | | | | |
| | $\frac{7 \dots 0}{E_0}$ | | $\frac{7 \dots 0}{S_0}$ | |
| | $\frac{15 \dots 8}{E_1}$ | | $\frac{15 \dots 8}{S_1}$ | |
| ent | $\frac{23 \dots 16}{E_2}$ | | $\frac{23 \dots 16}{S_2}$ | |
| | $\frac{31 \dots 24}{E_3}$ | | $\frac{31 \dots 24}{S_3}$ | |
| | | | | sal |

Seguidamente se muestra un esqueleto del proceso.

En la siguiente pregunta se utiliza el diseño ubicado en el directorio PROC_SERIE.

2. Utilizando los directorios ubicados en el directorio PROC_SERIE_fmte/ENSAMBLADO puede verificarse el funcionamiento del procesador cuando se incluye el formateador diseñado. Notemos que un programa usualmente sólo comprueba algunas operaciones. Efectúe esta comprobación después de responder a la última pregunta, una vez haya leído el mecanismo para efectuar una simulación ejecutando un programa..

Para ello, añada un proceso al programa de prueba (prueba_proc_MD_MI.vhd). Este proceso debe utilizar únicamente las señales de control opALU (“Figura 4.99” en la página 290), opMD (“Figura 4.107” en la página 292) y opSEC (“Figura 4.110” en la página 293) para determinar el tipo de instrucción interpretada en cada ciclo. Recuerde la forma de referenciar objetos en un diseño jerárquico utilizando referencias jerárquicas (Apéndice 3.5)⁴. Adjunte el código vhdL del proceso.

| | | | |
|---|--|---|--|
| Instrucciones ejecutadas | | Instrucciones aritmético-lógicas | |
| Instrucciones Load | | Instrucciones Store | |
| Instrucciones secuenciamiento condicional | | Instrucciones secuenciamiento incondicional | |

Seguidamente se muestra un esqueleto del proceso.

```

Proceso para obtener estadísticas
estadistiques: process (reloj)
variable ninst: natural;
variable v_opALUd: st_opALU;
...
begin
    if reloj'event and reloj = '1' then
        if Pcero = '1' then
            ninst := 0;
            ...
        else
            -- el atributo "delayed" produce una señal que es una copia de la señal en el instante "tiempo actual (now) - t (0 fs)" que se indica.
            v_opALUd := s_opALU'delayed(0 fs); -- la señal s_opALU se obtiene mediante una referencia jerárquica.
            ...
        end if;
        if parar = '1' then
            report "Instruccions Executades: " & integer'image(ninst)
            ...;
        end if;
    end if;
end process;

```

3. En la distribución del código se han utilizado “enlaces simbólicos”. Enlazan con los ficheros correspondientes del programa euclides. Otra alternativa es copiar el fichero.
4. Analice las definidas en el programa de prueba.

